AUTOMATIC GRADING PROGRAMS

BY

GEORGE E. FORSYTHE and NIKLAUS WIRTH

TECHNICAL REPORT CS17

FEBRUARY 12, 1965

COMPUTER SCIENCE DEPARTMENT
School of Humanities and Sciences
STANFORD UNIVERSITY

AUTOMATIC GRADING PROGRAMS */

by

George E. Forsythe and Niklaus Wirth **/

Abstract

The ALGOL grader programs are presented for the computer evaluation of student ALGOL programs. One is for a beginner's program; it furnishes random data and checks answers. The other provides a searching test of the reliability and efficiency of a rootfinding procedure. There is a statement of the essential properties of a computer system, in order that grader programs can be effectively used.

---

*/ Reproduction in Whole or in Part is permitted for any Purpose of the United States government. This report was supported in part by Office of Naval Research Contract Nonr-225(37) (NR 044-211) at Stanford University.

**/ Presented to the national meeting of the Association for Computing Machinery, Philadelphia, 27 August 1964, under the title "Automatic machine grading programs". The Stanford computers used for program tests are partially supported by the National Science Foundation under Grant No. GP-[illegible]8.

Discussion

In connection with introductory programming and numerical analysis courses at Stanford University, grading programs have been used intermittently since 1961. Our programs furnish data, check student performance in various ways, sometimes keep track of running time, and keep a "grade book" for the problems.

The Stanford routines are written separately for each problem. The most flexible and useful system for elementary classes was used with the Burroughs 220 computer in the BALGOL language, a dialect of ALGOL 58, and will be described first. Each grader program was written as a BALGOL-language procedure. It was then compiled together with a procedure called BUTTERFLY, written by Roger Moore. The result was a relocatable machine-language procedure, with a mechanism for equating its variables to variables of any BALGOL program, in just the form of the BALGOL compiler's own machine-language library procedures (SIN, WRITE, READ, etc.). Finally, the grader program was added to the compiler library tape for the duration of its use.

The use of a powerful algebraic language and system made it easy for an instructor to write grader programs with sophistication appropriate to the problem. The student needed only furnish one or two procedure statements to call a grader; we often furnished him cards to decrease the chance of error. Since each grader was precompiled on the library tape, little time was lost in adding it to each student's program at compile time. A simple handcoded mechanism made it easy for the operator to rescue a program from a run-time loop and send it into the

next case without dismissing the program entirely. It was possible to have several different graders concurrently on the library tape, to take care of different classes.

This powerful grading mechanism was possible only because of the BALGOL compiler with its own compiler-with-library generator, and because BUTTERFLY could generate relocatable machine-language programs. Neither our IBM 7090 BALGOL system nor our Burroughs B5000 ALGOL system has been so well adapted to the grading problem, to our regret, and we have had to make do with less desirable expedients. What these newer systems lack is an easy means of producing a machine-language library procedure which is fully equivalent to a source-language procedure with several parameters.

We recommend grading programs to all who teach programming and numerical analysis to masses of students, but the prospective user should first carefully investigate the systems available to him.

We give below a typical grading program, GRADER2, which was used in connection with an early problem in a beginning programming course. GRADER2 is suitable for grading a student's program which is written as a block. It has been translated into ALGOL and is given as part of a block containing one or more student programs.

In ALGOL there is a practical difficulty in putting many beginner programs into the same block with GRADER2--a syntactical error in just one student program may upset the compiler and prevent the testing of any program. It would be better if we could have GRADER2 precompiled into a system library and called separately by each student.

The student's problem is this: Given the integer N $(0 \leq N < 10)$ and the real array elements A[0], ...,A[N], B[0], ..., B[N], to write a program which makes MAX the maximum of A[0], ..., A[N] and which computes the numbers C[0], ..., C[2 × N] defined by the polynomial multiplication

$$\sum_{k=0}^{2N} C[k]t^k \equiv \left(\sum_{k=0}^{N} A[k]t^k\right) \cdot \left(\sum_{k=0}^{N} B[k]t^k\right) .$$

The student is told that he must arrange his program in the form:

```
    begin
        <all declarations>
         GRADER2 (<student number>, 1, N, MAX, A, B, C, START, FIN);
START:  <all statements of his solution>
         GRADER2  (<student number>, 2, N, MAX, A, B, C, START, FIN);
FIN:
    end;
```

Note that the subprocedure SET UP of GRADER2 goes to great trouble to be sure that no student will get the same data at different times, and that no two students will get the same data. This was intended to be sure that a student could not get correct answers from GRADER2 on one run, and use them for another run. We doubt the value of such precautions.

Observe that GRADER2 evaluates the correctness of the student answers, but in no way evaluates the running speed of the program nor the amount of storage used. This is appropriate for a beginning student of programming.

A more advanced student should have his performance examined more critically. As an example of this, we give a second grader program called Test, to be used in a numerical analysis class whose members can already program in ALGOL.

The procedure Test listed below is designed to examine rootfinding procedures. The students are asked to write an ALGOL procedure which finds an approximation to a 'root' x (i.e., point of sign-change) of a (not necessarily continuous) function f in the interval [a, b]. To be precise, x is any number such that

$$f(x_0) \leq 0 \equiv f(x_1) \geq 0 \qquad \text{(in the sense of ALGOL 60)}$$

and

$$a \leq x_0 \leq x \leq x_1 \leq b$$

and

$$|x_0 - x_1| \leq \epsilon \quad ,$$

where f, a, b, and ϵ are given parameters. Such an x always exists, if $f(a) \leq 0 \equiv f(b) \geq 0$. Each student is asked to submit an ALGOL program containing his procedure declaration and a single statement of the form

Test (rootfinder, 'student name')

As with GRADER2, all submitted programs are then enclosed together in an outer block, whose head contains the declaration of the procedure Test. Thus no use of library tapes is required. The block structure of ALGOL plays the very important role that all identifiers used by the student, including the name of his rootfinder, are strictly local to his program. They can therefore be chosen freely and cannot interfere

with identifiers in any other program. Nor can an identifier of the procedure Test interfere with any student's program. Of course this technique requires that the contributions be syntactically correct, but this is considered to be the minimum requirement for acceptance of a program from students at this level.

In order to obtain an estimate of the quality of the programs, one would like to know the accuracy of the answers, the number of evaluations of the function f it took to find the root (with the possibility of terminating the test, if a limit is exceeded), and perhaps also the time it took to find the root. It is furthermore desirable to check that the limits a and b and the tolerance ϵ were not changed during a test (this might occur, for example, if these parameters were called by name instead of by value), and whether the argument of f always remained within the interval $[a, b]$.

The following description of the procedure Test explains the methods of achieving these goals within the framework of an ALGOL program.

The declaration of Test contains the following variable-declarations:

grade represents the student's grade; it is cumulated during execution of several partial tests of one student's program.

m denotes his number of successful tests.

x is a real variable used as abscissa for the evaluation of f.

t records the time spent by the rootfinder.

A procedure P declared inside Test is the heart of the entire grader. The body of Test contains a series of calls of P; each call of P contains as actual parameters the data for one test example. E.g., the

call

$$P(0,2,{}_{10}{-5},1,1 - x, \underline{\text{true}}, 20, 1)$$

would cause the testing of the student's program with the function f: $f(x) = 1 - x$ in the interval $a = 0$, $b = 2$, with a desired accuracy of ${}_{10}{-5}$. The expected result (=1) is the fourth parameter to P. The sixth parameter indicates that a solution exists, the seventh is a limit for the number of evaluations of f, and the eighth indicates the number of the test case.

The procedure P subsequently calls the rootfinder (which is a formal parameter to Test) with the given parameters as data. However, P does not furnish the function f directly to the rootfinder, but rather substitutes a function procedure Q, which is declared in the head of P. Each call of Q then serves to increment the counter of calls of f and is also used to examine whether the argument of f lies within the prescribed interval.

The grader program has been used on the 7090 computer. In order to measure the time spent by the rootfinder and to recover from a possible error in the logic of the student's program, two code procedures have been introduced which cannot be described in ALGOL:

procedure Setime (n,L); integer n; label L;

initializes the core-clock to trap after n msec and to transfer control to L. Also,

procedure Reset (t); integer t;

disables the trap and assigns to t the number of msec spent since initialization of the clock. These procedures protect the entire grading run from failure due to one particular examinee's inability to solve a certain test problem.

The authors feel that particular emphasis should be put not only on the efficiency of the student's contribution, but also on its reliability. The choice of the test data reflects the possibilities of this grading method, since "wildly behaving" functions are used which are not likely to be foreseen by a careless programmer.

The program Test is believed to mark a further step in the automation of grading. Whereas GRADER2 bases its grade only on the binary answer "correct" or "wrong", Test also evaluates a program's quality i.e., reliability and effectiveness. It thus relieves the teacher from long and tedious grading work. Last, but not least, the machine may be more objective in grading than the human, because of its notable lack of prejudice and its inability to become bored.

The authors wish to acknowledge valuable suggestions made by A. J. Perlis and P. Naur in regard to grader programs.

An elementary grader program

```
begin
   procedure GRADER2 (STUDENT, ENTRY, N, MAX, A, B, C, START, FIN);
      value STUDENT, ENTRY; integer STUDENT, ENTRY, N; real MAX;
      real array A, B, C; label START, FIN;
      comment We assume the existence of a library real procedure
         TIME which produces the time of day as an integer in the
         interval [0, 2359];
   begin
      real procedure RANDOM;
      begin
         comment The value of RANDOM at each call is a different
            pseudo-random number from a flat distribution in the
            interval [0, 1]. The body is not written here ... ;
      end RANDOM;
      own real array CC[0:10];
      real S; own real MMAX;
      integer B3, B4, J, K;
      own integer TALLY, B1, B2;
      switch JUMP := L1, L2;
      procedure SET UP (CASE, N);
         value CASE; integer CASE, N;
      comment SET UP furnishes the student's data, which depend on
         the student's number, the time of day, and a pseudo-random
         number. SET UP also solves the case for the use of EVALUATE;
```

```
begin
   for K := 0 step 1 until N do
begin
   A[K] := RANDOM + (STUDENT + TIME) × 10-4;
   B[K] := RANDOM × sign(RANDOM - 0.5)
end;
comment  Now the student is messaged on the line-printer what
   data have been generated for him;
outstring (1, ‘FOR ␣ CASE’); outinteger (1, CASE);
outstring (1, ‘GRADER2 ␣ FURNISHES ␣ STUDENT’); outinteger(1,
   STUDENT); outstring (1,‘THE ␣ FOLLOWING ␣ DATA: ␣ ␣ A ␣ IS’);
for K := 0 step 1 until N do outreal (1, A[K]);
outstring (1, ‘B ␣ IS’);
for K := 0 step 1 until N do outreal (1, B[K]);
comment  Now GRADER2 solves the student's case for itself.
   GRADER2 does not use A[K] or B[K] for any values of  K
   outside 0 ≤ K ≤ N;
MMAX := A[0];
for K := 1 step 1 until N do
   if A[K] > MMAX then MMAX := A[K];
for K := 0 step 1 until N do
begin
   S := 0;
   for J := 0 step 1 until K do S := S + A[J] × B[K-J];
   CC[K] := S
end;
for K := N+1 step 1 until 2×N do
```

```
    begin
       S := 0;
       for J := K-N step 1 until N do S := S + A[J] × B[K-J];
       CC[K] := S
    end;
    comment  Now SET UP has solved the problem, and we exit
       to START, the entry to the student's solution.  The
       next call of GRADER2 will bring us back to EVALUATE;
    TALLY := TALLY + 1;
    go to START;
end SET UP;

procedure EVALUATE (CASE, N);
    value CASE; integer CASE, N;
begin
    B3 := 1;
    comment  EVALUATE examines the student's answers, writing
       them and its own answers, with comments on the student's
       performance, all on the line-printer;
    outstring (1, ‘FOR␣CASE’); outinteger (1, CASE);
    outstring (1, ‘STUDENT’); outinteger (1, STUDENT);
    outstring (1, ‘COMPUTES␣C␣TO␣BE’);
    for J := 0 step 1 until 2 × N do outreal (1, C[J]);
    outstring (1, ‘GRADER2␣COMPUTES␣C␣TO␣BE’);
    for J := 0 step 1 until 2 × N do outreal (1, CC[J]);
    for K := 0 step 1 until 2 × N do
       if abs(C[K] - CC[K]) > 10⁻⁴ then B3 := 0;
```

```
    if B3 = 1 then outstring (1, 'C␣IS␣ACCEPTABLE')
       else outstring (1, 'C␣IS␣NOT␣ACCEPTABLE');
    comment  A large tolerance was allowed for possible
       differences in the solutions because of different
       rounding off;
    B4 := 0;
    if MAX = MMAX then
    begin
       B4 := 1;
       outstring (1, 'MAX␣IS␣CORRECT')
    end
    else outstring (1, 'MAX␣IS␣INCORRECT');
  end EVALUATE;

  comment  Now come the statements of GRADER2 itself;
  if ENTRY = 1 then TALLY := 0 else go to JUMP[TALLY];
  comment  On the first call of GRADER2 by each student,
    ENTRY is 1.  On the later calls it is 2.  This and
    TALLY provide the mechanism for permitting different
    entries to GRADER2 on different calls;
  N := 5;
  SET UP (1, 5);
L1: EVALUATE (1, 5);
  B1 := B3; B2 := B4;
  N := 4;
  SET UP (2, 4);
```

```
L2: EVALUATE (2,4);
   comment  Case 2 is now complete, and GRADER2 punches a
      card for the "grade book";
   for K := STUDENT, 2, B1, B2, B3, B4 do outinteger (2, K);
   comment  GRADER2 now summarizes the situation for the
      student's line-printer listing;
   outstring (1, ‘STUDENT’); outinteger (1, STUDENT);
   outstring (1, ‘SCORES’); outinteger (1, B1+B2+B3+B4);
   outstring (1, ‘OUT␣OF␣4␣ON␣PROBLEM␣2.␣␣IF␣SCORE␣IS␣
      LESS␣THAN␣4,␣PLEASE␣SUBMIT␣REVISED␣PROGRAM␣LATER.’);
   comment  Now the program exits to the conclusion of the
      student's solution;
   go to FIN
end GRADER2;
comment  Now follow the students' programs;
begin
   comment  Here, for example, is the program for student
      number 515, with its calls on GRADER2;
   real array A, B, C [0:25];
   real S, MAX;
   integer J, K, N;
   GRADER2 (515, 1, N, MAX, A, B, C, START, FINISH);
START: for J := N+1 step 1 until 25 do A[J] := B[J] := 0;
   MAX := A[N];
   for K := N-1 step -1 until 0 do
      if MAX < A[K] then MAX := A[K];
```

```
        for K := 0 step 1 until 2×N do
        begin
           S := 0;
           for J := 0 step 1 until K do S := S + A[J]×B[K-J];
           C[K] := S
        end;
        GRADER2 (515, 2, N, MAX, A, B, C, START, FINISH);
     FINISH:
     end program of student 515;
     begin
        comment  Program of another student  ... ;
     end
     .....
  end tests of all student programs
```

An advanced grader program

```
begin comment  grader program for root-finding procedures;
procedure Test (Rootfinder, Name); procedure Rootfinder; string Name;
begin real x;  integer m, grade, time;
   procedure P (low, up, eps, root, f, is root, interation limit,
   problem no);
      value low, up, eps, root, is root, iteration limit, problem no;
      real low, up, eps, root, f;
      integer iteration limit, problem no;
      Boolean is root;
      begin real low 1, up 1, eps 1, root 1;
         integer n, t; Boolean is root 1;
         procedure Setime (n, L); integer n; label L; code;
         procedure Reset (t); integer t; code;
         procedure error (text, charge);
            value charge; integer charge; string text;
            begin outstring (text); grade := grade - charge;
               if grade ≤ 0 then
                  begin outstring (‘grade = 0’);
                     go to T exit
                  end
            end error;
         real procedure Q(y); value y; real y;
            begin if y < low 1 ∨ y > up 1 then
                  begin error (‘Argument␣outside␣interval’, 10);
                     Reset(t); go to P exit
```

```
            end;
        n := n + 1; if n > iteration limit then
            begin error (‘Convergence ␣ is ␣ too ␣ slow’, 5);
                Reset(t); go to P exit
            end;
        x := y; Q := f
    end Q;
low 1 := low; up 1 := up; eps 1 := eps; n := 0;
outstring (‘problem ␣no. =’); outreal (problem no);
Setime(100, fail);
Rootfinder (low 1, up 1, eps 1, Q, root 1, is root 1);
Reset(t); time := time + t;
if low 1 ≠ low ∨ up 1 ≠ up then
        error (‘boundary ␣ was ␣ altered’, 3);
if eps 1 ≠ eps then error (‘tolerance ␣ was ␣ altered’, 5);
if ¬ is root then
    begin if is root 1 then
        error (‘solution ␣ found ␣ where ␣ none ␣ exists’, 5)
        else begin outstring (‘correct ␣ reaction ␣ for ␣ no ␣ root’);
                m := m + 1
            end
    end
    else
    begin real tol; tol := abs (root - root 1);
        if tol > eps then error (‘incorrect ␣ root’, 5)
            else begin outstring (‘correct ␣ root ␣ found’); m := m+1
                end;
```

```
                    outstring ('no.␣of␣iterations ='); outreal (n);
              end;
              go to P exit;
        fail: error ('failure', 10);
        P exit:
        end P;
        m := 0; grade := 100; time := 0; outstring (Name);
        P(-2, 2, ₁₀-6, -1, x+1, true, 10, 1);
        P(-1, 1, ₁₀-6, -1, x+1, true, 10, 2);
        P(-1, 1, ₁₀-6, 1, x-1, true, 10, 3);
        P(2, 5, ₁₀-6, 0, x-1, false, 10, 4);
        P(-2, 3.5, ₁₀-5, 2, x↑3 - x × 3 - 2-₁₀-20, true, 30, 5);
        P(₁₀-3, 99.9, ₁₀-5, 0.01, x + 1/x - 100.01, true, 50, 6);
        P(-1, 2, ₁₀-5, 0, sign(x), true, 30, 7);
        P(-3, 100, ₁₀-4, 0, exp(-x) -1, true, 50, 8);
        P(0, 20, ₁₀-4, 0.95, (x + 0.05)↑ 0.1 - 1, true, 30, 9);
        P(0, 100, ₁₀-5, 1, if x < 1 then x-1 else ₁₀-10, true, 30, 10);
        P(-2.4, 4.2, ₁₀-4, 3, ((((x-3) × x+5) × x-15) × x+4) × x-12,
           true, 50, 11);
        P(5₁₀-3, 1, 5₁₀-3, 0.0265, if x  < 0.02122 then -1 else
           if x > 0.03183 then 1 else cos(1/x), true, 30, 12);
        outstring ('end␣of␣test.␣No. ␣of␣correct␣problems =');
        outreal (m); outstring ('time ='); outreal (time);
        outstring ('grade ='); outreal (grade);
  T exit:
  end T;
```

```
comment Subsequently follow the students' programs, each containing
a procedure declaration and a call of Test enclosed in a block;
begin
   procedure Bisect (x0, x1, tol, func, result, is result);
   real x0, x1, tol, result;
   real procedure func;
   Boolean is result;
   begin ............
   ...........
   end Bisect;
   Test (Bisect, ‘Tom␣Jones’)
end;
............
comment further students' programs follow here;
............
end Grader program
```

STANFORD UNIVERSITY

TECHNICAL REPORTS DISTRIBUTION LIST

CONTRACT Nonr-225(37)

(NR 044-211)

ARMED SERVICES TECHNICAL
INFORMATION AGENCY
ARLINGTON HALL STATION
ARLINGTON 12, VIRGINIA 10

CHIEF OF NAVAL RESEARCH
CODE 432
OFFICE OF NAVAL RESEARCH
WASHINGTON 25, D.C. 2

COMMANDING OFFICER
OFFICE OF NAVAL RESEARCH
BRANCH OFFICE
1000 GEARY ST.
SAN FRANCISCO 9, CALIF. 1

COMMANDING OFFICER
OFFICE OF NAVAL RESEARCH
BRANCH OFFICE
495 SUMMER STREET
BOSTON 10, MASSACHUSETTS 1

COMMANDING OFFICER
OFFICE OF NAVAL RESEARCH
BRANCH OFFICE 346 BROADWAY
NEW YORK 13, NEW YORK 1

COMMANDING OFFICER
OFFICE OF NAVAL RESEARCH
BRANCH OFFICE
NAVY NO. 100, FLEET POST OFFICE
NEW YORK, NEW YORK 2

COMMANDING OFFICER
OFFICE OF NAVAL RESEARCH
BRANCH OFFICE
1030 EAST GREEN ST.
PASADENA 1, CALIF. 1

ARGONNE NATIONAL LABORATORY
TECHNICAL INFORMATION DIVISION
9700 SOUTH CASS AVE.
ARGONNE, ILLINOIS 1

COMMANDING OFFICER
U.S. NAVAL ELECTRONICS LAB.
ATTN. R. F. ARENZ
F. A. SABRANSKY
SAN DIEGO 52, CALIF. 2

COMPUTATION CENTER
UNIVERSITY OF TEXAS
AUSTIN 12, TEXAS 1

COMPUTER REVIEWS
ROOM 5-C
211 E. 43rd STREET
NEW YORK 36, NEW YORK 1

DEPARTMENT OF MATHEMATICS
HARVARD UNIVERSITY
CAMBRIDGE, MASSACHUSETTS 1

DEPARTMENT OF MATHEMATICS
MASSACHUSETTS INST. OF TECH
CAMBRIDGE, MASSACHUSETTS 1

SPACE TECHNOLOGY LABS.
ONE SPACE PARK
ATTN. IRENE MATHEWS
REDONDO BEACH, CALIF. 1

INSTITUTE FOR MATHEMATICAL
SCIENCES
NEW YORK UNIVERSITY
NEW YORK, 3, NEW YORK 1

LIBRARY
DIGITAL COMPUTER LAB.
UNIVERSITY OF ILLINOIS
URBANA, ILLINOIS 1

LIBRARIAN, COMMUNICATIONS
RESEARCH DIVISION
INSTITUTE FOR DEFENCE ANALYSES
VON NEUMANN HALL
PRINCETON, NEW JERSEY 1

NATIONAL AERONAUTICS AND
SPACE ADM.
LEWIS RESEARCH CENTER
21000 BROOKPARK ROAD
ATTN. LIBRARY
CLEVELAND 35, OHIO 1

OFFICE OF TECHNICAL SERVICES
DEPARTMENT OF COMMERCE
WASHINGTON 25, D. C. 1

DOCUMENT CENTER
STANFORD RESEARCH INSTITUTE
333 RAVENSWOOD AVE.
MENLO PARK, CALIF. 1

TECHNICAL INFORMATION OFFICER
NAVAL RESEARCH LABORATORY
WASHINGTON 25, D. C. 6

THE MITRE CORP. LIBRARY
ATTN. SUPERVISOR LIBRARY
SERVICES
BEDFORD, MASSACHUSETTS 2

UNIVERSITY OF CALIFORNIA
ATTN. LIBRARY,
NUMBERICAL ANALYSIS RES.
LOS ANGELES 24, CALIF. 1

UNIVERSITY OF MIAMI
SCIENTIFIC COMPUTING CENTER
P.O. BOX 8011
CORAL GABLES 46, FLORIDA 1

U.S. NAVAL ORDNANCE LABORATORY
WHITE OAK
ATTN: MATHEMATICS DEPARTMENT
SILVER SPRING, MARYLAND 1

U. S. NAVAL ORDNANCE LABORATORY
WHITE OAK
ATTN. LIBRARY
SILVER SPRING, MARYLAND 1

U. S. NAVY MANAGEMENT OFFICE
DEPARTMENT OF THE NAVY
WASHINGTON 25, D. C. 1

U. S. NAVAL WEAPONS LABORATORY
DAHLGREN, VIRGINIA 1

PROFESSOR W. F. ATCHISON
RICH ELECTRONIC COMPUTER CENTER
GEORGIA INSTITUTE OF TECHNOLOGY
ATLANTA, GEORGIA 1

DR. ERWIN H. BAREISS
APPLIED MATHEMATICS DIVISION
ARGONNE NATIONAL LABORATORY
9700 SOUTH CASS AVE.
ARGONNE, ILLINOIS 1

PROF. ROBERT C. F. BARTELS
COMPUTING CENTER
UNIVERSITY OF MICHIGAN
ANN ARBOR, MICHIGAN 1

DR. RICHARD BELLMAN
THE RAND CORPORATION
1700 MAIN ST.
SANTA MONICA, CALIFORNIA 1

PROF. GARRETT BIRKHOFF
MATHEMATICS DEPARTMENT
HARVARD UNIVERSITY
CAMBRIDGE 38, MASSACHUSETTS 1

BERNARD H. BISSINGER
DEPARTMENT OF MATHEMATICS
MATH LIBRARY
LEBANON VALLEY COLLEGE
ANNVILLE, PENNSYLVANIA 1

MR. TRUMAN BOTTS
DEPARTMENT OF MATHEMATICS
UNIVERSITY OF VIRGINIA
CHARLOTTESVILLE, VIRGINIA 1

PROF. JAMES BRAMBLE
INSTITUTE FLUID DYNAMICS AND
APPLIED MATHEMATICS
UNIVERSITY OF MARYLAND
COLLEGE PARK, MARYLAND 1

PROF. A. T. BRAUER
DEPARTMENT OF MATHEMATICS
UNIVERSITY OF NORTH CAROLINA
CHAPEL HILL, NORTH CAROLINA 1

PROF. GEORGE W. BROWN
WESTERN DATA PROCESSING CENTER
UNIVERSITY OF CALIF.
LOS ANGELES 24, CALIF. 1

PROF. JOHN W. CARR, III
MOORE SCHOOL OF ELE. ENGR.
UNIVERSITY OF PENNSYLVANIA
PHILDELPHIA 4, PENN. 1

MR. ROBERT L. CAUSEY
LOCKHEED AIRCRAFT CORP.
MISSILES AND SPACE
3251 HANOVER ST.
PALO ALTO, CALIF. 1

PROF. ABRAHAM CHARNES
THE TECHNOLOGICAL INSTITUTE
NORTHWESTERN UNIVERSITY
EVANSTON, ILLINOIS 1

PROF. B. A. CHARTRES
DIVISION OF APPLIED MATHEMATICS
BROWN UNIVERSITY
PROVIDENCE 12, RHODE ISLAND 1

PROF. HARVEY COHN
DEPARTMENT OF MATHEMATICS
UNIVERSITY OF ARIZONA
TUSCON 25, ARIZONA 1

PROF. STEPHEN CRANDALL
DEPARTMENT OF MECHANICS
MASS. INST. OF TECHNOLOGY
CAMBRIDGE 39, MASSACHUSETTS 1

PROF. PHILIP DAVIS
DEPARTMENT OF MATHEMATICS
BROWN UNIVERSITY
PROVIDENCE 12, RHODE ISLAND 1

PROF. DAVID B. DEKKER
DIRECTOR, RESEARCH COMPUTER LAB.
UNIVERSITY OF WASHINGTON
SEATTLE 5, WASHINGTON 1

PROF. R. J. DEVOGELAERE
DEPARTMENT OF MATHEMATICS
UNIVERSITY OF CALIFORNIA
BERKELEY 4, CALIF. 1

PROF. C. L. DOLPH
311 PARK AVENUE
ANN ARBOR, MICHIGAN 1

PROF. JIM DOUGLAS, JR.
DEPARTMENT OF MATHEMATICS
THE RICE INSTITUTE
HOUSTON 1, TEXAS 1

PROF. R. J. DUFFIN
MATHEMATICS DEPARTMENT
CARNEGIE INSTITUTE OF TECHNOLOGY
PITTSBURGH 13, PENNSYLVANIA 1

DR. ROBERT DICKSON
LOCKHEED AIRCRAFT CORPORATION
MISSILES AND SPACE DIVISION
3251 HANOVER STREET
PALO ALTO, CALIF. 1

PROF. PAUL DWYER
403 LENAWEE DRIVE
ANN ARBOR, MICHIGAN 1

MR. THOMAS A. DWYER
DIRECTOR, COMPUTER SCIENCE
DEPARTMENT OF PHYSICS
UNIVERSITY OF DAYTON
DAYTON 9, OHIO 1

MR. EDGAR L. EICHHORN
BURROUGHS CORPORATION
ELECTRADATA FACILITY
460 SIERRA MADRE VILLA
PASADENA, CALIF. 1

MR. FRANK ENGEL, JR.
ANALYTICAL DEPARTMENT 4-L-39
WESTINGHOUSE ELECTRIC CORP.
700 BRADDOCK AVENUE
EAST PITTSBURGH, PENNSYLVANIA 1

DR. G. W. EVANS, II
STANFORD RESEARCH INSTITUTE
333 RAVENSWOOD AVENUE
MENLO PARK, CALIF. 1

PROF. F. A. FICKEN
DEPARTMENT OF MATHEMATICS
NEW YORK UNIVERSITY
UNIVERSITY HEIGHTS
NEW YORK 53, NEW YORK 1

MR. WERNER FRANK
4617 LOARKWOOD AVENUE
WOODLAND HILLS, CALIF. 1

PROF. J. N. FRANKLIN
CALIFORNIA INST. OF TECHNOLOGY
PASADENA 4, CALIF. 1

PROF. BERNARD FRIEDMAN
DEPARTMENT OF MATHEMATICS
UNIVERSITY OF CALIFORNIA
BERKELEY 4, CALIF. 1

DR. H. H. GERMOND
RCA SERVICE COMPANY
BLDG. 989-1 MU 811
PARTRICK AIR FORCE BASE
FLORIDA 1

PROF. WALLACE GIVENS
DEPARTMENT OF ENGR. SCIENCE
NORTHWESTERN UNIVERSITY
EVANSTON, ILLINOIS 1

DR. H. H. GOLDSTINE
RESEARCH CENTER
INTERNATIONAL BUSINESS MACHINES
YORKTOWN, NEW YORK 1

PROF. SAUL GORN
MOORE SCHOOL OF ELEC. ENGINEERING
UNIVERSITY OF PENNSYLVANIA
PHILADELPHIA 4, PENNSYLVANIA 1

PROF. L. GREENBERG
DEPARTMENT OF MATHEMATICS
BROWN UNIVERSITY
PROVIDENCE, RHODE ISLAND 1

MR. SAMUEL W. GREENHOUSE
CHIEF, SECTION OF STAT. AND MATH
NATIONAL INSTITUTE OF MENTAL HEALTH
BETHESDA 14, MARYLAND 1

PROF. R. T. GREGORY
COMPUTATION CENTER
UNIVERSITY OF TEXAS
AUSTIN 12, TEXAS 1

DR. T. N. E. GREVILLE
MATH. RESEARCH CENTER, U.S. ARMY
UNIVERSITY OF WISCONSIN
MADISON 6, WISCONSIN 53706 1

PROF. PRESTON HAMMER
DIRECTOR, NUMERICAL ANALYSIS LAB.
UNIVERSITY OF WISCONSIN
MADISON 6, WISCONSIN 1

DR. ELDON HANSEN
347 ALDEAN
MOUNTAIN VIEW, CALIF. 1

PROF. T. J. HIGGINS
DIRECTOR, UNIVERSITY OF WISCONSIN
ENGINEERING EXPERIMENT STATION
MADISON 6, WISCONSIN 1

PROF. T. W. HILDEBRANDT
DEPARTMENT OF MATHEMATICS
OHIO STATE UNIVERSITY
COLUMBUS, OHIO 1

DR. WILLIAM HOOKER
PHILCO CORPORATION
3875 FABIAN WAY
PALO ALTO, CALIF. 1

DR. A. S. HOUSEHOLDER
OAK RIDGE NATIONAL LAB.
P. O. BOX X
OAK RIDGE, TENNESSEE 1

PROF. B. E. HUBBARD
INSTITUTE FLUID DYNAMICS AND
APPLIED MATHEMATICS
UNIVESITY OF MARYLAND
COLLEGE PARK, MARYLAND 1

DR. M. A. HYMAN
IBM SYSTEMS DEVELOPMENT CENTER
7220 WISCONSIN AVE.
BETHESDA, MARYLAND 1

PROF. EUGENE ISAACSON
INSTITUTE OF MATHEMATICAL SCIEN.
NEW YORK UNIVERSITY
4 WASHINGTON PLACE
NEW YORK 3, NEW YORK 1

DR. W. J. JAMESON
RESEARCH DIVISION
COLLINS RADIO COMPANY
CEDAR RAPIDS, IOWA 1

PROF. M. KATZ
DEPARTMENT OF MATHEMATICS
UNIVERSITY OF WISCONSIN
MILWAUKEE 11, WISCONSIN 1

PROF. THOMAS A. KEENAN
COMPUTING CENTER
UNIVERSITY OF ROCHESTER
ROCHESTER 20, NEW YORK 1

PROF. WILLIAM B. KEHL
COMPUTATION AND DATA PROC. CENT.
825 CATHEDRAL OF LEARNING
UNIVERSITY OF PITTSBURGH
PITTSBURGH 13, PENNSYLVANIA 1

DR. ERVAND G. KOGBETLIANTZ
ROCKEFELLER INST. FOR MEDICAL
RESEARCH
66th ST. AND YORK AVE.
NEW YORK, NEW YORK 1

PROF. R. E. LANGER
DEPARTMENT OF MATHEMATICS
UNIVERSITY OF WISCONSIN
MADISON 6, WISCONSIN 1

PROF. RALPH E. LEE
DIRECTOR OF COMPUTER CENTER
SCHOOL OF MINES AND METALLURGY
UNIVERSITY OF MISSOURI
ROLLA, MISSOURI 1

PROF. ARVID LONSETH
DEPARTMENT OF MATHEMATICS
OREGON STATE COLLEGE
CORVALLIS, OREGON 1

MR. B. MITTMAN
ARMOUR RESEARCH FOUNDATION
10 WEST 35TH STREET
CHICAGO 16, ILLINOIS 1

PROF. JAMES M. MOORE
DEPARTMENT OF INDUSTRIAL
ENGINEERING
NORTHEASTERN UNIVERSITY
BOSTON 12, MASSACHUSETTS 1

PROF. T. S. MOTZKIN
DEPARTMENT OF MATHEMATICS
UNIVERSITY OF CALIFORNIA
BERKELEY 4, CALIF. 1

DR. J. P. NASH
LOCKHEED MISSILE AND SPACE CORP.
SUNNYVALE, CALIF. 1

PROF. C. D. OLDS
P. O. BOX 462
LOS ALTOS, CALIFORNIA 1

DR. JAMES M. ORTEGA
DIVISION 5426
SANDIA CORPORATION
ALBUQUERQUE, NEW MEXICO 1

DR. JOSEPH OTTERMAN
P. O. BOX 2008
UNIVERSITY OF MICHIGAN
ANN ARBOR, MICHIGAN 1

B. N. PARLETT
INST. FOR MATHEMATICAL SCIENCES
4 WASHINGTON PLACE
NEW YORK 3, NEW YORK 1

DR. SEYMOUR V. PARTER
DEPARTMENT OF MATHEMATICS
UNIVERSITY OF WISCONSIN
MADISON, WISCONSIN 1

PROF. ALAN J. PERLIS
COMPUTATION CENTER
CARNEGIE INSTITUTE OF TECHNOLOGY
PITTSBURGH 13, PENNSYLVANIA 1

PROF. CLAY L. PERRY
DIRECTOR, COMPUTATION CENTER
UNIVERSITY OF CALIFORNIA
SCHOOL OF SCIENCE AND ENGINEERING
LA JOLLA, CALIF. 1

PROF. MURRAY PROTTER
DEPARTMENT OF MATHEMATICS
UNIVERSITY OF CALIFORNIA
BERKELEY 4, CALIF. 1

PROF. WERNER RHEINBOLDT
COMPUTER SCIENCE CENTER
UNIVERSITY OF MARYLAND
COLLEGE PARK, MARYLAND 1

PROF. R. D. RICHTMYER
AEC COMPUTING AND APPLIED
MATHEMATICS CENTER
INSTITUTE OF MATHEMATICAL SCIENCES
NEW YORK UNIVERSITY
4 WASHINGTON PLACE
NEW YORK 3, NEW YORK 1

MRS. MAXINE ROCKHOFF
DEPARTMENT OF COMMERCE
NATIONAL BUREAU OF STANDARDS
WASHINGTON 25, D. C. 1

EDWARD M. ROSEN
SENIOR MATHEMATICIAN
APPLIED MATHEMATICS DEPARTMENT
MONSANTO CHEMICAL CO.
800 N. LINDBERGH
ST. LOUIS 66, MISSOURI 1

DR. SAMUEL SCHECHTER
RESEARCH SCIENTIST
AEC COMPUTING CENTER
NEW YORK UNIVERSITY
NEW YORK 3, NEW YORK 1

PROF. LOWELL SCHOENFELD
MATHEMATICS DEPARTMENT
PENNSYLVANIA STATE UNIVERSITY
STATE COLLEGE, PENNSYLVANIA 1

DR. DANIEL SHANKS
DAVID TAYLOR MODEL BASIN
WASHINGTON 7, D. C. 1

JAMES M. SLOSS
MATHEMATICS DEPARTMENT
UNIVERSITY OF CALIFORNIA
SANTA BARBARA, CALIF. 1

PROF. THOMAS H. SOUTHARD
CHAIRMAN, MATHEMATICS DEPARTMENT
ALAMEDA STATE COLLEGE
22300 FOOTHILL BLVD.
HAYWARD, CALIFORNIA 1

PROF. M. L. STEIN
DEPARTMENT OF MATHEMATICS
INSTITUTE OF TECHNOLOGY
UNIVERSITY OF MINNESOTA
MINNEAPOLIS 4, MINNESOTA 1

PROF. J. J. STOKER
INSTITUTE OF MATHEMATICAL SCIENCES
25 WAVERLY PLACE
NEW YORK 3, NEW YORK 1

MR. H. C. THACHER, JR.
ARGONNE NATIONAL LABORATORY
9700 SOUTH CASS AVENUE
ARGONNE, ILLINOIS 1

MR. ROLAND THOMPSON
BURROUGHS CORPORATION
6071 2ND AVE.
DETROIT 32, MICHIGAN 1

DR. GENE THOMPSON
UNITED TECHNOLOGY CORP.
1050 E. ARQUES AVE.
SUNNYVALE, CALIF. 1

PROF. JOHN TODD
MATHEMATICS DEPARTMENT
CALIF. INST. OF TECHNOLOGY
PASADENA 4, CALIFORNIA 1

PROF. RICHARD S. VARGA
DEPARTMENT OF MATHEMATICS
CASE INSTITUTE OF TECHNOLOGY
UNIVERSITY CIRCLE
CLEVELAND 6, OHIO 1

PROF. WOLFGANG WASOW
DEPARTMENT OF MATHEMATICS
UNIVERSITY OF WISCONSIN
MADISON 6, WISCONSIN 1

PROF. M. G. WURTELE
DEPARTMENT OF METEOROLOGY
UNIVERSITY OF CALIFORNIA
LOS ANGELES 24, CALIF. 1

PROF. DAVID M. YOUNG, JR.
DEPARTMENT OF MATHEMATICS
UNIVERSITY OF TEXAS
AUSTIN 12, TEXAS 1

DISTRIBUTION VIA ONR LONDON

F. C. ADEY F. L. A., CHIEF LIBRARIAN
LEICESTER COLLEGE OF ART AND TECH.
THE NEWARKES,
LEICESTER ENGLAND 1

MATHEMATISCH CENTRUM
2E BOERHAAVESTRAAT 49
AMSTERDAM O, NETHERLANDS 1

PROF. A. C. AITKEN
MATHEMATICAL INSTITUTE
UNIVERSITY OF EDINBURGH
16 CHAMBERS STREET
EDINBURGH 1, SCOTLAND 1

DR. ENZO APARO
INSTITUTO NAZIONALE PER LE
APPLICAZIONI DEL CALCOLO
PIAZZALE DELL SCIENZE 7
ROMA, ITALY 1

PROF. DR. CHARLES BLANC
CENTRE DE CALCUL ELECTRONIQUE
AVENUE DE COUR, 33
LAUSANNE, SWITZERLAND 1

JEAN A. BAUDOT
COMPUTING CENTRE
DEPARTMENT OF MATHEMATICS
UNIVERSITE DE MONTREAL
MONTREAL, CANADA 1

PROF. DR. F. L. BAUER
MATHEMATISCHES INST. DER
TECHNISCHEN
HOCHSCHULE
8 MUNCHEN 2
ARCISSTRASSE 21
GERMANY 1

MR. NIELS IVAR BECH
REGNECENTRALEN
G1. CARLSBERGVEJ 2
VALBY, DENMARK 1

J. G. BYRNE
ENGINEERING SCHOOL
TRINITY COLLEGE
DUBLIN 2, IRELAND 1

DR. STIG COMET
INTERNATIONAL COMPUTATION CENTRE
PALAZZO DEGLI UFFICI
ZONA DELL'E. U. R.
ROMA, ITALY 1

PROF. DR. L. COLLATZ
INSTITUT FUR ANGEWANDTE MATH
UNIVERSITAT HAMBURG
(24A) HAMBURG 13
ROTHENBAUMCHAUSSEE 67/69
GERMANY 1

PROF. GERMUND G. DAHLQUIST
ROYAL INST. OF TECHNOLOGY
STOCKHOLM DIV. OF APPLIED MATH
STOCKHOLM 70, SWEDEN 1

DR. LESLIE FOX
OXFORD UNIVERSITY
OXFORD, ENGLAND 1

DR. CARL-ERIK FROBERG
ADV. FOR NUMERISK ANALYSIS
LUNDS UNIVERSITET
LUND, SWEDEN 1

MR. RICHARD GOODMAN
DIRECTOR, AUTOMATIC PROGRAMMING
INFORMATION CENTER
DEPARTMENT OF MATHEMATICS
THE TECHNICAL COLLEGE
BRIGHTON, ENGLAND 1

DR. J. M. HAMMERSLEY
INSTITUTE OF STATISTICS
ST. CROSS ROAD
OXFORD, ENGLAND 1

PROF. DR. PETER HENRICI
LEHRSTUHL FUR HOHERE MATHEMATIK
EIGENOSSISCHE TECHNISCHE
HOCHSCHULE
ZURICH, SWITZERLAND 1

DR. JOSEPH HERSCH
E. T. H. ZIMMER 20D
ZURICH 6, SWITZERLAND 1

INTERNATIONAL COMPUTATION CENTER
PALAZZO DEGLI UFFICI
ZONA DELL'E. U. R.
ROMA, ITALY 1

PROF. J. KUNTZMANN
LABORATOIRE DE CALCUL
UNIVERSITE DE GRENOBLE
44-46 AVENUE FELIX-VIALLET
GRENOBLE, ISERE
FRANCE 1

PROF. PENTTI LAASONEN
FINLAND INSTITUTE OF TECHNOLOGY
HELSINKI, FINLAND 1

PROF. C. LANCZOS
INSTITUTE FOR ADVANCED STUDIES
64-65 MERRION SQUARE
DUBLIN, IRELAND 1

PROF. J. L. LIONS
DEPARTMENT DE MATHEMATIQUES
UNIVERSITE DE PARIS
11 RUE PIERRE CURIE
PARIS 5E, FRANCE 1

PROF. L MIRSKY
THE UNIVERSITY
SHEFFIELD, ENGLAND 1

DR. A. R. MITCHELL
MATHEMATICS DEPARTMENT
ST. SALVATOR'S COLLEGE
UNIVERSITY OF ST. ANDREWS
ST. ANDREWS, FIFE, SCOTLAND 1

MR. JEAN A. MUSSARD
EXECUTIVE SECRETARY
PROVISIONAL INTERNATIONAL
COMPUTATION CENTER
PALAZZO DEGLI UFFICI
ZONA DELL'E. U. R.
ROMA, ITALY 1

PROF. A. OSTROWSKI
CERTENAGO-MONTAGNOLA
BEI LUGANO
SWITZERLAND

PROF. E. S. PAGE
DIRECTOR, UNIVERSITY COMPUTING LAB.
NEWCASTLE ON TYNE
ENGLAND 1

PROF. DR. R. SAUER
MATHEMATISCHES INSTITUT
DER TECHNISCHEN HOCHSCHULE
MUNCHEN 2
ARCISSTR. 21, GERMANY 1

PROF. E. STIEFEL
INSTITUT FUR ANGEWANDTE MATH.
EIDG. TECHNISCHE HOCHSCHULE
ZURICH, SWITZERLAND 1

PROF. DR. HEINZ UNGER
INSTITUT FUR ANGEWANDTE MATH.
DER UNIVERSITAT BONN
WEGELERSTRASSE 10
BONN, GERMANY 1

PROF. DR. A. WALTHER
DIREKTOR, INSTITUT FUR PRAKTISCHE
MATHEMATIK
TECHNISCHE HOCHSCHULE
(16) DARMSTADT, GERMANY 1

DR. S. B. L. WILSON
COMPUTING DEPARTMENT
AERO ENGINE DIVISION
ROLLS-ROYCE LIMITED
ELTON ROAD
DERBY, ENGLAND 1

PROF. RIGAL
SECRETAIRE A L'AFCAL
48 RUE DES POITIERS
MONTBELIARD, DOUBS
FRANCE 1

PROF. DR. H. W. WIELANDT
MATHEMATISCHES INSTITUT
UNIVERSITAT TUBINGEN
TUBINGEN (14B)
WILHELMSTRASSE 7, GERMANY 1

DR. M. V. WILKES
UNIVERSITY MATHEMATICAL LAB.
CORN EXCHANGE STREET
CAMBRIDGE, ENGLAND 1

DR. J. H. WILKINSON
MATHEMATICS DIVISION
NATIONAL PHYSICAL LAB.
TEDDINGTON, MIDDLESEX
ENGLAND 1

HONORARY LIBRARIAN, BRITISH COMPUTER
SOCIETY,
FINSBURY COURT, FINSBURY PAVEMENT
LONDON E. C. 2, ENGLAND 1

THE LIBRARY, BUILDING 465
ATOMIC ENERGY RES. ESTABLISHMENT
HARWELL, DIDCOT
BERKSHIRE, ENGLAND 1

JEAN A BAUDOT
COMPUTING CENTER
DEPARTMENT OF MATHEMATICS
UNIVERSITE DE MONTREAL
MONTREAL, CANADA 1

OTHER FOREIGN ADDRESSES

DR. JOHN M. BENNETT
ADOLPH BASSER COMPUTER LAB.
UNIVERSITY OF SYDNEY
SYDNEY, N.S.W.
AUSTRALIA 1

W. A. COWAN, LIBRARIAN
BARR SMITH LIBRARY
THE UNIVERSITY
ADELAIDE
SOUTH AUSTRALIA 1

PROF. A. GONZALEZ DOMINGUEZ
THE INSTITUTE OF CALCULUS
UNIVERSITY OF BUENOS AIRES
AVENIDA DE MAYO 760, 20 PISO
BUENOS AIRES, ARGENTINA 1

FACULTAD DE CIENCIAS EXACTAS
Y NATURAL
BIBLIOTECA SECCION CANJE
CASILLA DE CORREO 1766
BUENOS AIRES
REPUBLICA ARGENTINA 1

PROF. SANTIAGO FRIEDMANN
DIRECTOR DEL CENTRO DE COMPUTACION
UNIVERSIDAD DE CHILE
CASILLA 2777
SANTIAGO, CHILE 1

PROF. ALAN GIBB
DEPARTMENT OF MATHEMATICS
UNIVERSITY OF ALBERTA
CALBARY, ALBERTA
CANADA 1

DR. C. C. GOTLIEB
COMPUTATION CENTER
MCLENNAN LABORATORY
UNIVERSITY OF TORONTO
TORONTO, CANADA 1

DR. J. F. HART
HEAD, COMPUTING CENTRE
THE UNIVERSITY OF WESTERN ONTARIO
LONDON, CANADA 1

PROF. THOMAS E. HULL
COMPUTATION CENTER
MCLENNAN LABORATORY
UNIVERSITY OF TORONTO
TORONTO, CANADA 1

PROF. W. KAHAN
DEPARTMENT OF MATHEMATICS
UNIVERSITY OF TORONTO
TORONTO, CANADA 1

PROF. J. MCNAMEE
DEPARTMENT OF MATHEMATICS
UNIVERSITY OF ALBERTA
ALBERTA, CANADA 1

PROF. MANUEL SADOSKY
INSTITUTO DE CALCULO
FACULTAD DE CIENCIAS EXACTAS
Y NATURALES
PERU 27, BUENOS AIRES
REPUBLICA ARGENTINA 1

PROF. H. SCHWERDTFEGER
DEPARTMENT OF MATHEMATICS
MCGILL UNIVERSITY
MONTREAL, QUEBEC
CANADA 1

MR. J. C. VELLA
PUBLISHING SECRETARY
SOCIEDAD ARGENTINA DE CALCULO
AVENIDO DE MAYO 760 - 20 PISO
BUENOS AIRE,
ARGENTINA 2

PROF. R. KODIS
DEPARTMENT OF MATHEMATICS
BROWN UNIVERSITY
PROVIDENCE, RHODE ISLAND 1

PROF. C. DONALD LA BUDDE
DEPARTMENT OF CHEMISTRY
NEW YORK UNIVERSITY
UNIVERSITY HEIGHTS
NEW YORK 53, NEW YORK 1

ADDITIONAL COPIES FOR FUTURE
REQUIREMENTS 50

CONTRACT NONR-225(37)
NOVEMBER 1963
247

UNCLASS

Security Classification

DOCUMENT CONTROL DATA - R&D

(Security classification of title, body of abstract and indexing annotation must be entered when the overall report is classified)

1. ORIGINATING ACTIVITY (Corporate author) Computer Science Department Stanford University Stanford, Calif. 94305	2a. REPORT SECURITY CLASSIFICATION UNCLASS 2b. GROUP —

3. REPORT TITLE

AUTOMATIC GRADING PROGRAMS

4. DESCRIPTIVE NOTES (Type of report and inclusive dates)

Manuscript for publication (Technical Report)

5. AUTHOR(S) (Last name, first name, initial)

FORSYTHE, George E. and WIRTH, Niklaus

6. REPORT DATE	7a. TOTAL NO. OF PAGES	7b. NO. OF REFS
February 12, 1965	17	none

8a. CONTRACT OR GRANT NO. Nonr-225(37) b. PROJECT NO. NR-044-211 c. d.	9a. ORIGINATOR'S REPORT NUMBER(S) CS17 9b. OTHER REPORT NO(S) (Any other numbers that may be assigned this report) none

10. AVAILABILITY/LIMITATION NOTICES

1. Qualified requesters may obtain copies of this report from: Defence Document Center, Cameron Station Alexandria, Virginia.
2. Chief, Input Section, Clearinghouse for Federal Scientific & Tech. Information, CFSTI Sills Bldg. 5285 Port Royal Road, Springfield, Virginia 22151.

11. SUPPLEMENTARY NOTES	12. SPONSORING MILITARY ACTIVITY
	OFFICE OF NAVAL RESEARCH Code 432

13. ABSTRACT

The ALGOL grader programs are presented for the computer evaluation of student ALGOL programs. One is for a beginner's program; it furnishes random data and checks answers. The other provides a searching test of the reliability and efficiency of a rootfinding procedure. There is a statement of the essential properties of a computer system, in order that grader programs can be effectively used.

DD FORM 1 JAN 64 1473

UNCLASSIFIED

Security Classification

14 KEY WORDS	LINK A ROLE	LINK A WT	LINK B ROLE	LINK B WT	LINK C ROLE	LINK C WT
1. Grading by Computers						
2. Teaching by Computers						
3. ALGOL Programs for Grading						
4. Teaching Machines						

INSTRUCTIONS

1. ORIGINATING ACTIVITY: Enter the name and address of the contractor, subcontractor, grantee, Department of Defense activity or other organization *(corporate author)* issuing the report.

2a. REPORT SECURITY CLASSIFICATION: Enter the overall security classification of the report. Indicate whether "Restricted Data" is included. Marking is to be in accordance with appropriate security regulations.

2b. GROUP: Automatic downgrading is specified in DoD Directive 5200.10 and Armed Forces Industrial Manual. Enter the group number. Also, when applicable, show that optional markings have been used for Group 3 and Group 4 as authorized.

3. REPORT TITLE: Enter the complete report title in all capital letters. Titles in all cases should be unclassified. If a meaningful title cannot be selected without classification, show title classification in all capitals in parenthesis immediately following the title.

4. DESCRIPTIVE NOTES: If appropriate, enter the type of report, e.g., interim, progress, summary, annual, or final. Give the inclusive dates when a specific reporting period is covered.

5. AUTHOR(S): Enter the name(s) of author(s) as shown on or in the report. Enter last name, first name, middle initial. If military, show rank and branch of service. The name of the principal author is an absolute minimum requirement.

6. REPORT DATE: Enter the date of the report as day, month, year; or month, year. If more than one date appears on the report, use date of publication.

7a. TOTAL NUMBER OF PAGES: The total page count should follow normal pagination procedures, i.e., enter the number of pages containing information.

7b. NUMBER OF REFERENCES: Enter the total number of references cited in the report.

8a. CONTRACT OR GRANT NUMBER: If appropriate, enter the applicable number of the contract or grant under which the report was written.

8b, 8c, & 8d. PROJECT NUMBER: Enter the appropriate military department identification, such as project number, subproject number, system numbers, task number, etc.

9a. ORIGINATOR'S REPORT NUMBER(S): Enter the official report number by which the document will be identified and controlled by the originating activity. This number must be unique to this report.

9b. OTHER REPORT NUMBER(S): If the report has been assigned any other report numbers *(either by the originator or by the sponsor)*, also enter this number(s).

10. AVAILABILITY/LIMITATION NOTICES: Enter any limitations on further dissemination of the report, other than those imposed by security classification, using standard statements such as:

(1) "Qualified requesters may obtain copies of this report from DDC."

(2) "Foreign announcement and dissemination of this report by DDC is not authorized."

(3) "U. S. Government agencies may obtain copies of this report directly from DDC. Other qualified DDC users shall request through ________________."

(4) "U. S. military agencies may obtain copies of this report directly from DDC. Other qualified users shall request through ________________."

(5) "All distribution of this report is controlled. Qualified DDC users shall request through ________________."

If the report has been furnished to the Office of Technical Services, Department of Commerce, for sale to the public, indicate this fact and enter the price, if known.

11. SUPPLEMENTARY NOTES: Use for additional explanatory notes.

12. SPONSORING MILITARY ACTIVITY: Enter the name of the departmental project office or laboratory sponsoring *(paying for)* the research and development. Include address.

13. ABSTRACT: Enter an abstract giving a brief and factual summary of the document indicative of the report, even though it may also appear elsewhere in the body of the technical report. If additional space is required, a continuation sheet shall be attached.

It is highly desirable that the abstract of classified reports be unclassified. Each paragraph of the abstract shall end with an indication of the military security classification of the information in the paragraph, represented as *(TS)*, *(S)*, *(C)*, or *(U)*.

There is no limitation on the length of the abstract. However, the suggested length is from 150 to 225 words.

14. KEY WORDS: Key words are technically meaningful terms or short phrases that characterize a report and may be used as index entries for cataloging the report. Key words must be selected so that no security classification is required. Identifiers, such as equipment model designation, trade name, military project code name, geographic location, may be used as key words but will be followed by an indication of technical context. The assignment of links, roles, and weights is optional.